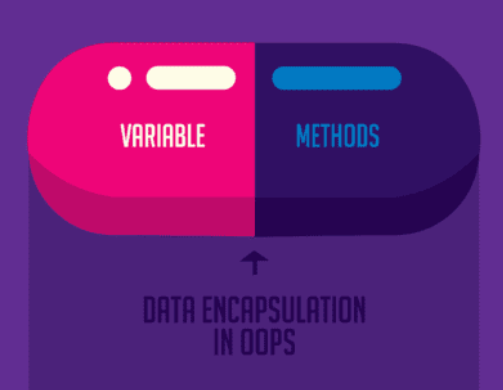
Encapsulation -Abstraction - 2022



By definition, encapsulation describes the idea of bundling data and methods that work on that data within one unit, like a class in Java. This concept is also often used to hide the internal representation, or state of an object from the outside. This is called information hiding.

The general idea of this mechanism is simple. For example, you have an attribute that is not visible from the outside of an object. You bundle it with methods that provide read or write access. Encapsulation allows you to hide specific information and control access to the internal state of the object.

Take an example of credit card where we can provide getter method for credit card number, but we may not provide the CVV number ie. no getter and setter method. It Java, it is specific to access modifiers like private, public, protected. It is a language’s mechanism for restricting access to some of the object’s components.

In his book on [object-oriented design](https://en.wikipedia.org/wiki/Object-oriented_design), [Grady Booch](https://en.wikipedia.org/wiki/Grady_Booch) defined encapsulation as "**the process of compartmentalizing the elements of an abstraction that constitute its structure and behavior**; encapsulation serves to separate the contractual interface of an abstraction and its implementation.

* Wrapping of data in a construct such that it hides its internal data
* Wrapping of internal data and function.
* Providing protection to the data members.
* Provide behaviour methods to control access of data in desired fashion
* Encapsulation also protects the integrity of the component, by preventing users from setting the internal data

**Abstraction**

Its main goal is to handle complexity by hiding unnecessary details from the user. That enables the user to implement more complex logic on top of the provided abstraction without understanding or even thinking about all the hidden complexity. Example will be mobile phone. Abstraction is a way to segregate implementation from specification. We do not know what happens inside a monitor and it does not matter also. So abstraction expose only the details which are concerned for the user. Abstraction is achieved using Abstract Class and Interface.

**Polymorphisam: An Object is in different forms and in each form its exhibit the same functionality but the implementation is different. A person can speak multiple language.**

**Difference between Abstraction and Encapsulation**

|  |  |
| --- | --- |
| **Abstraction** | **Encapsulation** |
| **1. Abstraction solves the problem in the design level.** | 1. **Encapsulation solves the problem in the implementation level**. |
| 2. **Abstraction is used for hiding the unwanted data and giving relevant data.** | 2. Encapsulation means hiding the code and data into a single unit to protect the data from outside world. |
| **3. Abstraction lets you focus on what the object does instead of how it does it** | 3. **Encapsulation means hiding the internal details or mechanics of how an object does something.** |
| 4. **Abstraction**- Outer layout, used in terms of design. For Example:-   Outer Look of a Mobile Phone, like it has a display screen and keypad buttons to dial a number. | 4. **Encapsulation**- **Inner layout**, used in terms of implementation.  For Example:- Inner Implementation detail of a Mobile Phone, how keypad button and Display Screen are connect with each other using circuits. |

* **Abstraction is accomplished by using Interface**
* **Encapsulation/Information/Data hiding is accomplished by using**

**Difference between Abstract class and Interface**

**Abstract Class**

An abstract class provides a base infrastructure. That is why abstract class has both abstract and non-abstract method. Java provides both **AbstractList** and **List** interface. Any class extending abstract class will get the default behaviour. Abstract class is like a franchise which provides base infrastructure.

Example: I want to take a franchise of manufacturing Volvo type vehicles called VolvoIndia. I will take the base setup from Volvo Sweden.

**public abstract class** VolvoVehicle {  
 **public void** start(Key key) {  
 key.apply();  
 }  
  
 **public void** stop(Key key) {   
 key.apply();  
 }  
  
 **public abstract void** provideSafety();  
 **public abstract void** provideLuxury();  
}

**public interface** Key {  
 **void** apply();  
}

**Implementation classes**

**public class** LuxuryBus **extends** VolvoVehicle {  
  
 @Override  
 **public void** provideSafety() {  
  
 }  
  
 @Override  
 **public void** provideLuxury() {  
  
 }  
}

**public class** EconomyBus **extends** VolvoVehicle {  
  
 @Override  
 **public void** provideSafety() {  
  
 }  
  
 @Override  
 **public void** provideLuxury() {  
  
 }  
}

In this case, Key is an interface. Now what will happen if we declare an interface instead of Abstract class. If we define an interface, it will become a contract or license from Volvo and we have to provide complete implementation from the scratch. We can have similar example from KFC, McDonald or Dommino Pizza. We are not supposed to prepare all kinds of recipe rather I can provide base infrastructure for recipe preparation. If we do not have skilled resources for chicken pizza preparation, we can take the base setup, all other recipe, we can override and we will put more spices to increase the taste. That is why you may not have heard that Dommino Pizza from Indira Nagar is better that that of Kormangla.

**Interface** 🡺 License

Interface is like a contract or license, where we get the license information but we have to make our own infrastructure and we have to provide complete implementation. Example will Government provides Bar license, it is upto the wine shop to implement in his own way. Similarly an author provides a copyright license to make a movie. But the producer and director can make the movies in their own way.

**public interface** Author {  
 **void** picturizeStory();  
}

**public interface** Government {  
 **void** sellLiquor();  
}

What will happen if we declare Government as abstract class? In this case all the liquor shop have to follow a standard norms which may their profit. If we define Author as an abstract class, I have to follow the complete guidelines and character in the movies which may not be popular. Take an example of movie **3 idiots**.